

# Massively Parallel Particle-in-Cell modeling with WarpX



Andrew Myers

Lawrence Berkeley National Laboratory

On behalf of the WarpX team (lead: Jean-Luc Vay @ LBNL)  
LBNL, LLNL, SLAC, CEA, DESY, Modern Electron, CERN

2022 NERSC GPUs for Science Day

virtual  
October 25th, 2022

*Note: this presentation includes pre-acceptance  
benchmarks of NERSC Perlmutter & OLCF Frontier.*



U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science

# WarpX: a growing Multidisciplinary, Multi-Institutional Team from research labs, academia and industry



U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science



**ATAP**

ACCELERATOR TECHNOLOGY &  
APPLIED PHYSICS DIVISION



**AMReX**

**NERSC**

**NESAP**



DESY (Germany)



Peter Scherpelz,  
Michael Kieburz,  
Kevin Zhu, Roelof  
E. Groenewald



Phil Miller



CERN (Switzerland)



CEA Saclay (France)

Henri  
Vincenti

Luca  
Fedeli

Antonin  
Sainte-Marie

Neil  
Zaim

Maxence  
Thévenet

Severin  
Diederichs

Alexander  
Sinn

Lorenzo  
Giacomel

+ a growing list of  
contributors

Jean-Luc Vay  
(ECP PI)



Marco Garten



Axel  
Huebl



Rémi  
Lehe



Chad  
Mitchell



Ji  
Qiang



Ryan  
Sandberg



Olga  
Shapoval



Yinjian  
Zhao



Edoardo  
Zoni



Ann Almgren  
(ECP coPI)



John  
Bell



Kevin  
Gott



Junmin  
Gu



Revathi  
Jambunathan



Hannah  
Klion



Prabhat  
Kumar



Andrew  
Myers



Weiquan  
Zhang



David Grote  
(ECP coPI)



(NESAP)

Marc Hogan  
(ECP coPI)



Lixin  
Ge



Cho  
Ng



**SLAC**

# Particle Accelerators are Essential Tools in Modern Life

## Medicine



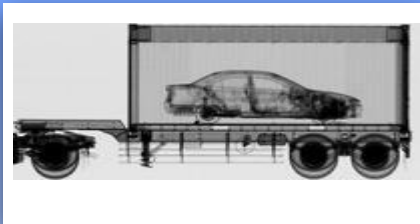
- ~9,000 medical accelerators in operation worldwide
- 10's of millions of patients treated
- 50 medical isotopes, routinely produced with accelerators

## Industry



- ~20,000 industrial

## National Security



- Cargo scanning

## Discovery Science



- ~30% of Nobel Prizes in Physics since 1939
- 4 Nobel Prizes in Physics for research at CERN

**Next generation of accelerators needs next generation of HPC modeling tools!**

- Irradiation
- Welding/cutting
- Annual value of all products that use accel. Tech.: \$500B

support of non-proliferation

facilities

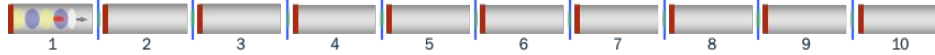
**Opportunity for much bigger impact by reducing size and cost.**

**Modeling: Exploration → Understanding → Design**

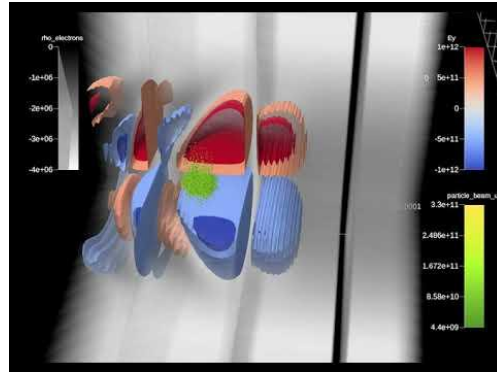
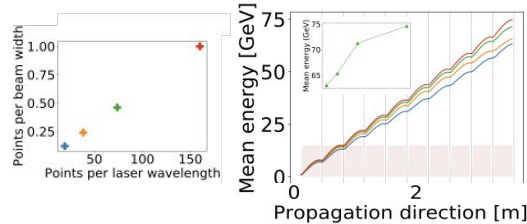


# Demonstrated 10 stages of LWFA modeling

WarpX demo is first 3D simulation of chain of 10 plasma accelerator stages



Convergence study scanned from low to high resolution on 3 to 768 GPUs/run.



In Situ Rendering:

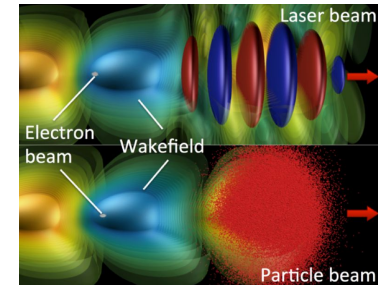
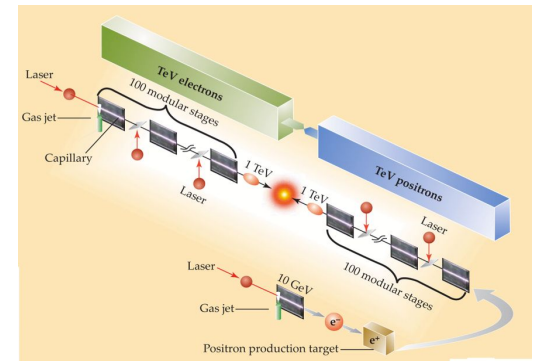
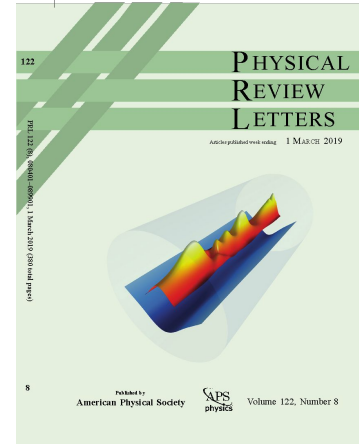
- GPU-accelerated
- zero-copy

Libraries:

Ascent + VTK-m

Schematic of a  
pot. laser-plasma  
1 TeV collider

2019 – acceleration to  
8 GeV in 20 cm.



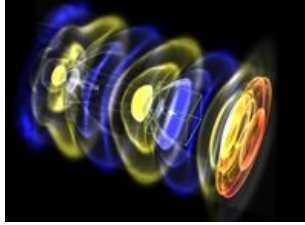
Plasma accelerator can be  
driven by laser or particle beam

- A multi-TeV plasma-based particle accelerator would be based on multiple plasma stages.
- Each stage is computationally-intensive.
- Need for  $\times 100$  stages  $\times 100$  ensemble.

Require most advanced algorithms + Exascale

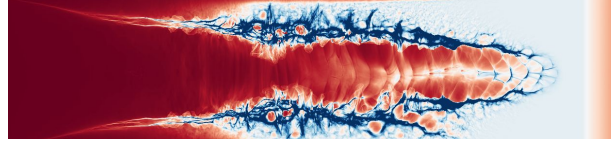
J.-L. Vay, A. Huebl, "Uses of In Situ/In Transit Methods in Large-Scale Modeling of Plasma-Based Particle Accelerators," ISAV'20 Workshop Keynote (2020); M. Larsen et al., "The ALPINE In Situ Infrastructure: Ascending from the Ashes of Strawman," ISAV'17 Conference Paper (2017)

# WarpX supports a growing number of applications

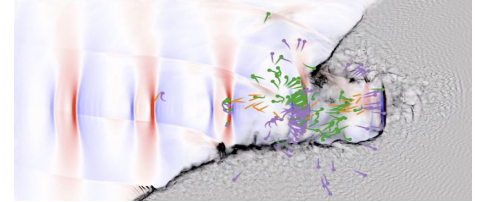


Plasma accelerators  
(LBNL, DESY, SLAC)

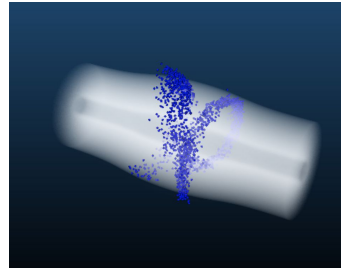
Laser-ion acceleration -  
advanced mechanisms (LBNL)



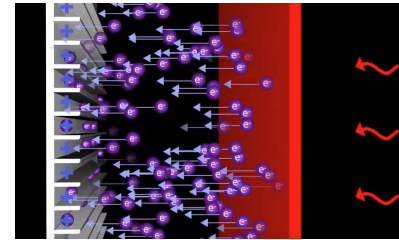
Plasma mirrors and high-field  
physics + QED (CEA  
Saclay/LBNL)



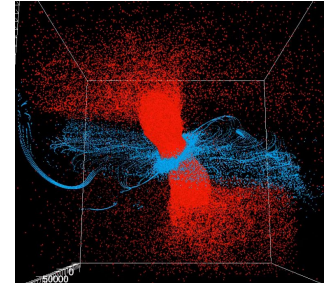
Plasma confinement,  
fusion devices  
(Zap Energy,  
Avalanche Energy)



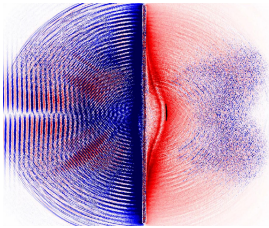
Thermionic converter  
(Modern Electron)



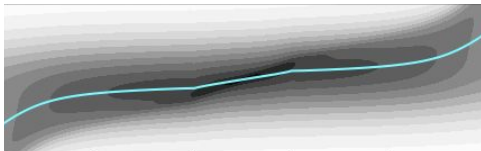
Pulsars, magnetic  
reconnection (LBNL)



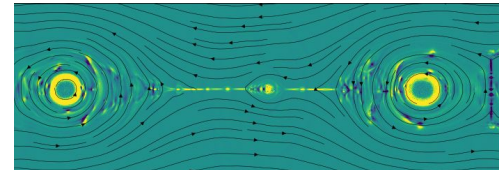
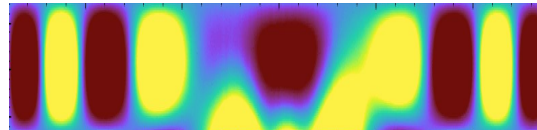
Laser-ion  
acceleration -  
laser pulse  
shaping (LLNL)



Magnetic fusion sheaths (LLNL)



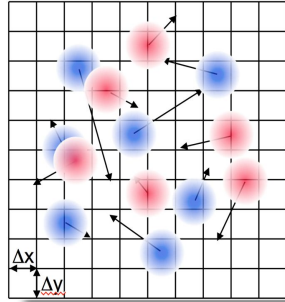
Microelectronics (LBNL) - ARTEMIS



# WarpX: Advanced Multi-Physics Particle-in-Cell Code for Exascale

## Particle-in-Cell

macro-  
particles  
  
electro-  
magnetic (EM)  
fields on a grid



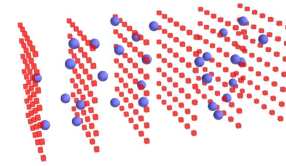
Advanced Algorithms Pioneered by our Team  
boosted frame, spectral solvers, Galilean  
frame, embedded boundaries + CAD, MR, ...

## Multi-Physics Modules (PICSAR)

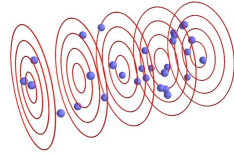
field ionization of atomic levels, Coulomb  
collisions, QED processes (e.g. pair creation),  
macroscopic materials

## Geometries

- 1D3V, 2D3V,  
3D3V and  
RZ (quasi-  
cylindrical)



3D Cartesian grid



Cylindrical grid (schematic)

## Multi-Node Parallelization

- MPI: 3D domain decomposition
- dynamic load balancing

## On-Node Parallelization

- GPU: CUDA, HIP and SYCL
- CPU: OpenMP

## Scalable, Parallel I/O

- AMReX plotfile and  
openPMD (HDF5 or ADIOS)
- in situ diagnostics

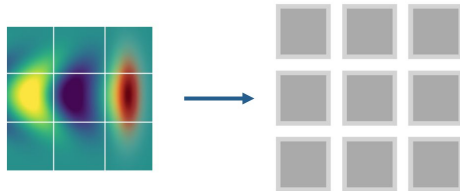


# Portable Performance through Exascale Programming Model

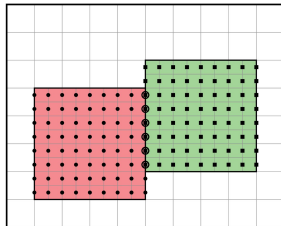
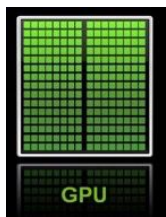
AMReX library



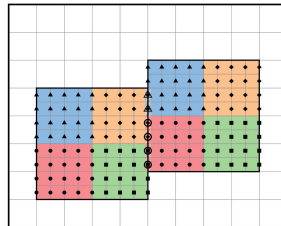
- Domain decomposition & MPI communications: MR & load balance



- Performance-Portability Layer: GPU/CPU/KNL



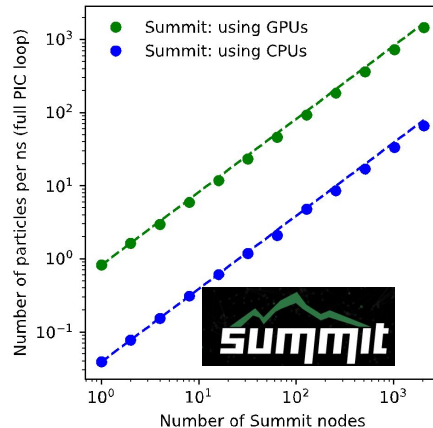
without tiling



with tiling



Data Structures

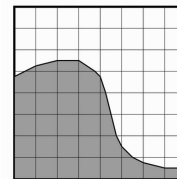


- Write the code once, specialize at **compile-time**

ParallelFor (/Scan/Reduce)

```
amrex::ParallelFor( n_particles,  
    [=] AMREX_GPU_DEVICE (long i) {  
  
    UpdatePosition( x[i], y[i], z[i],  
        ux[i], uy[i], uz[i], dt );  
  
});
```

- Parallel linear solvers (e.g. multi-grid Poisson solvers)
- Embedded boundaries
- Runtime parser for user-provided math expressions (incl. GPU)



# Porting to GPUs: Fortran -> C++

Yee FDTD update for Ey, Fortran 90:

```
subroutine push_electric_field_y(ylo, yhi, &
    ey, eylo, eyhi, bx, bxlo, bxhi, bz, bzlo, bzhi, &
    jy, jylo, jyhi, mudt, dtsdx, dtsdz) &
    bind(c,name='push_electric_field_y')

    use amrex_fort_module, only : amrex_real
    implicit none

    integer,          intent(in)    :: ylo(3), yhi(3)
    integer,          intent(in)    :: eylo(3), eyhi(3)
    integer,          intent(in)    :: bxlo(3), bxhi(3), bzlo(3), bzhi(3)
    integer,          intent(in)    :: jylo(3), jyhi(3)
    real(amrex_real), intent(inout) :: ey(eylo(1):eyhi(1), eylo(2):eyhi(2), eylo(3):eyhi(3))
    real(amrex_real), intent(in)    :: bx(bxlo(1):bxhi(1), bxlo(2):bxhi(2), bxlo(3):bxhi(3))
    real(amrex_real), intent(in)    :: bz(bzlo(1):bzhi(1), bzlo(2):bzhi(2), bzlo(3):bzhi(3))
    real(amrex_real), intent(in)    :: jy(jylo(1):jyhi(1), jylo(2):jyhi(2), jylo(3):jyhi(3))
    real(amrex_real), value        :: mudt, dtsdx, dtsdz

    integer :: j,k,l

    do l = 1, 3
        do k = 1, 3
            do j = 1, 3
                Ey(j,k,l) = Ey(j,k,l) - dtsdx * (Bz(j,k,l) - Bz(j-1,k,l)) &
                    + dtsdz * (Bx(j,k,l) - Bx(j,k,l-1)) &
                    - mudt * jy(j,k,l)
            end do
        end do
    end do
end subroutine push_electric_field_y
```

C++ version:

```
AMREX_GPU_HOST_DEVICE AMREX_FORCE_INLINE
void push_electric_field_y(int j, int k, int l,
    amrex::Array4<amrex::Real> > const& Ey,
    amrex::Array4<amrex::Real> const& Bx,
    amrex::Array4<amrex::Real> const& Bz,
    amrex::Array4<amrex::Real> const& jy,
    amrex::Real mudt, amrex::Real dtsdx, amrex::Real dtsdz)
{
    Ey(j,k,l) = Ey(j,k,l) - dtsdx * (Bz(j,k,l) - Bz(j-1,k,l))
        + dtsdz * (Bx(j,k,l) - Bx(j,k,l-1))
        - mudt * jy(j,k,l);
}
```

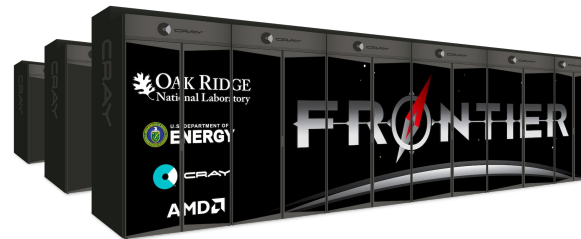
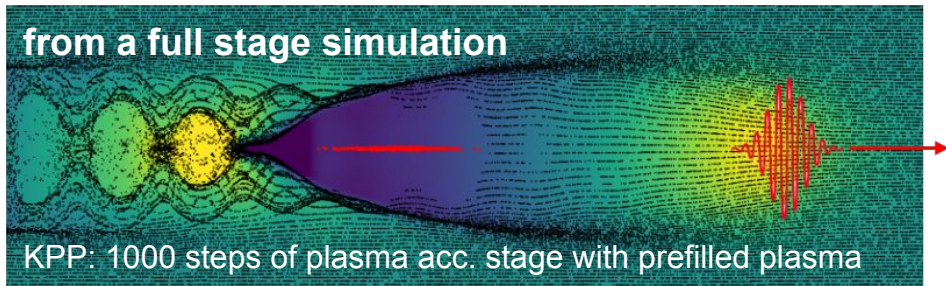
Array4 multidimensional array object  
facilitates scientific programming

Upfront cost in development time, but:

- Single codebase for NVIDIA, AMD, Intel
- C++ tends to get first priority in terms of compiler support
- Once made, porting to A100 + MI250X (relatively) easy



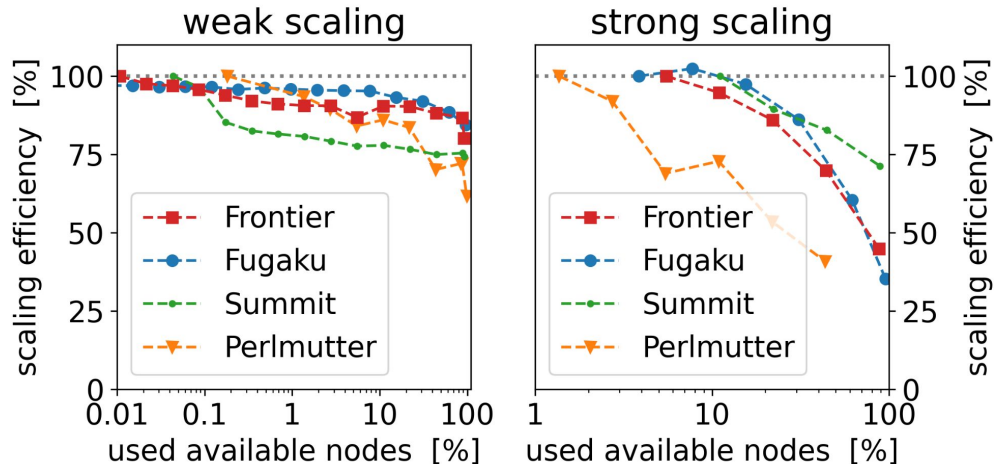
# KPP on Frontier: x500 over Baseline on Cori



*Note: Perlmutter & Frontier are pre-acceptance!*

Demonstrated scaling **4-5 orders** of magnitude

**Figure-of-Merit** over time

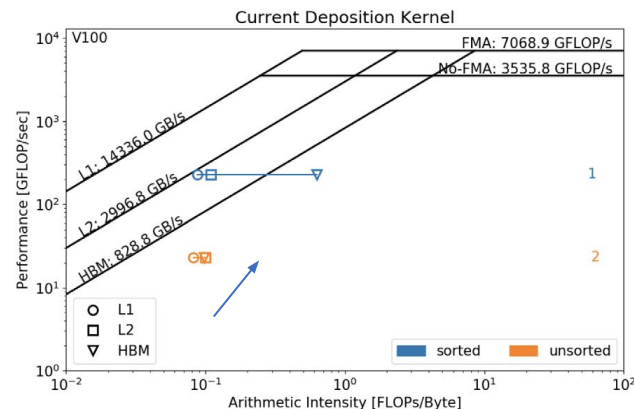
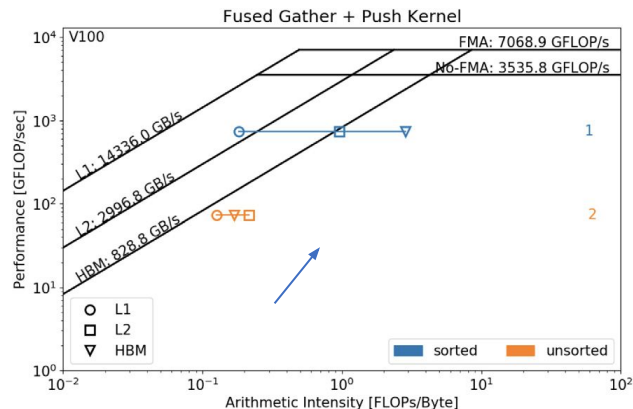


Date	Code	Machine	N <sub>c</sub> /Node	Nodes	FOM
3/19	WarpX	Cori	0.4e7	6 625	2.2e10
3/19	WarpX	Cori	0.4e7	6 625	1.0e11
6/19	WarpX	Summit	2.8e7	1 000	7.8e11
9/19	WarpX	Summit	2.3e7	2 560	6.8e11
1/20	WarpX	Summit	2.3e7	2 560	1.0e12
2/20	WarpX	Summit	2.5e7	4 263	1.2e12
6/20	WarpX	Summit	2.0e7	4 263	1.4e12
7/20	WarpX	Summit	2.0e8	4 263	2.5e12
3/21	WarpX	Summit	2.0e8	4 263	2.9e12
6/21	WarpX	Summit	2.0e8	4 263	2.7e12
7/21	WarpX	Perlmutter	2.7e8	960	1.1e12
12/21	WarpX	Summit	2.0e8	4 263	3.3e12
4/22	WarpX	Perlmutter	4.0e8	928	1.0e12
4/22	WarpX	Perlmutter†	4.0e8	928	1.4e12
4/22	WarpX	Summit	2.0e8	4 263	3.4e12
4/22	WarpX	Fugaku†	3.1e6	98 304	8.1e12
6/22	WarpX	Perlmutter	4.4e8	1 088	1.0e12
7/22	WarpX	Fugaku	3.1e6	98 304	2.2e12
7/22	WarpX	Fugaku†	3.1e6	152 064	9.3e12
7/22	WarpX	Frontier	8.1e8	8 576	1.1e13



# Kernel Tuning: Particle Sorting increases locality

Top kernels: **current deposition** (scatter) and **field gather + push**: improvement from particle *sorting*.



A Myers et al., **Porting WarpX to GPU-accelerated platforms.**  
*Parallel Computing.*, 108:102833 (2021). [DOI:10.1016/j.parco.2021.102833](https://doi.org/10.1016/j.parco.2021.102833)

Also have **shared memory** deposition algorithms that perform ~40% better on A100 (up to 3x better on MI250X)

# Transitioning to an Integrated Ecosystem



**WarpX**

full PIC,  
LPA/LPI

**HiPACE++**

quasi-static, PWFA

**ARTEMIS**

microelectronics

**ImpactX**

accelerator lattice  
design

Object-Level  
Python Bindings  
extensible, AI/ML

**pyAMReX**

**PIC SAR**  
QED Modules

**ABLASTR library: common PIC physics**

**AMReX**

Containers, Communication,  
Portability, Utilities

**Diagnostics**

I/O  
code coupling

**openPMD**

**ADI  
OS2**

**HD  
F5**

**ZFP**

**Asc  
ent**

**VTK  
-m**

**FFT**

on- or  
multi-  
device

**Lin.  
Alg.**

BLAS++  
LAPACK++

**MPI**

**CUDA, OpenMP, SYCL, HIP**

mac  
OS



Desktop  
to  
HPC

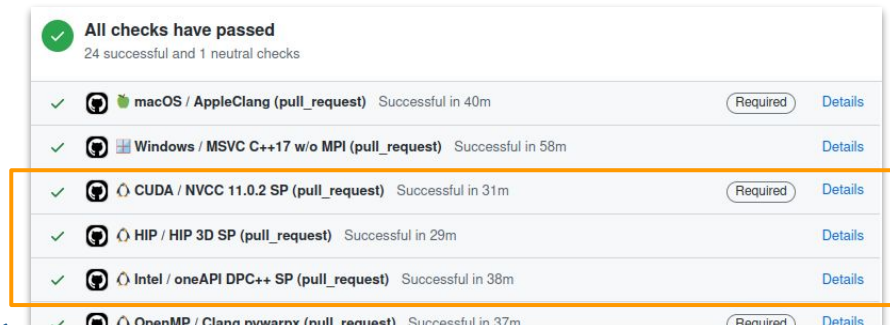


# An open interface with the community

Online Documentation:  
[warpx|hipace|impactx.readthedocs.io](http://warpx.hipace.impactx.readthedocs.io)



Open-Source Development & Benchmarks:  
[github.com/ECP-WarpX](https://github.com/ECP-WarpX)



Rapid and easy installation on any platform:



`python3 -m pip install .`



`brew tap ecp-warpX/warpX`  
`brew install warpX`



`conda install`  
`-c conda-forge warpX`



`spack install warpX`  
`spack install py-warpX`



`cmake -S . -B build`  
`cmake --build build --target install`



`module load warpX`  
`module load py-warpX`



EXASCALE  
COMPUTING  
PROJECT



LDR

D

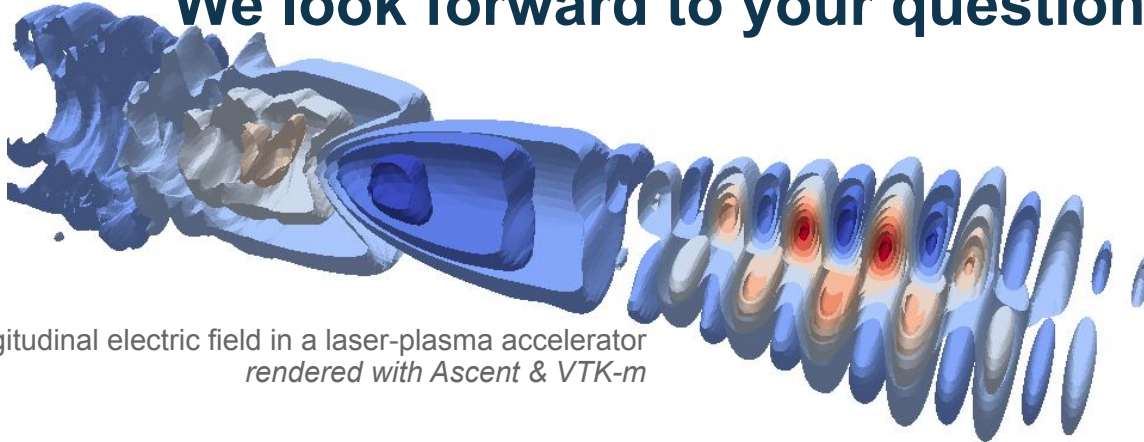


**188 physics benchmarks** run on every code change of WarpX  
**8 physics benchmarks + 32 tests** for ImpactX



# Thank you for your attention

## We look forward to your questions



WarpX: longitudinal electric field in a laser-plasma accelerator  
*rendered with Ascent & VTK-m*

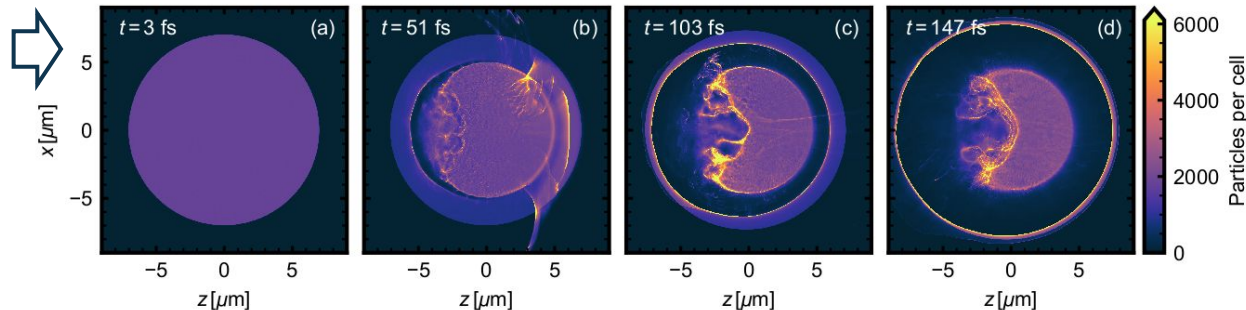


This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative. This work was supported by the Laboratory Directed Research and Development Program of Lawrence Berkeley National Laboratory under U.S. Department of Energy Contract No. DE-AC02-05CH11231. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725, the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231, and the supercomputer Fugaku provided by RIKEN.

# GPU Computing at Scale Requires Advanced Load Balancing

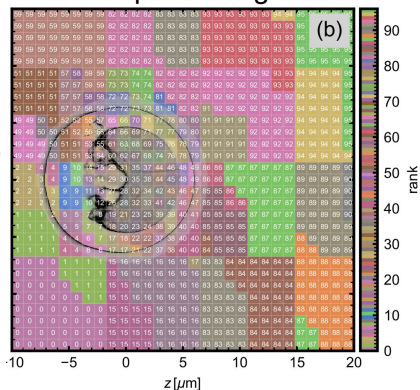
## Application Challenges

- Plasma Mirrors & Laser-Ion Acceleration: moving front
- Laser Wakefield Accelerator: Injected Beam Particles



domain decomposition example:

Z-order space filling curve

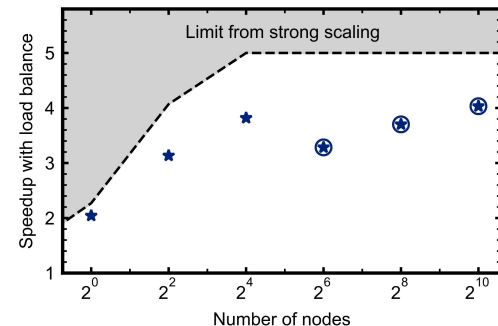


## In Situ Cost Analysis

- basis for distribution functions
- realistic cost: kernel timing

## Result: 3.8x speedup!

- production-quality, easy-to-use
- larger simulation: mitigate local memory spikes



M. Rowan, A. Huebl, K. Gott, R. Lehe, M. Thévenet, J. Deslippe, J.-L. Vay, "In-Situ Assessment of Device-Side Compute Work for Dynamic Load Balancing in a GPU-Accelerated PIC Code," PASC21, DOI:10.1145/3468267.3470614 (2021)

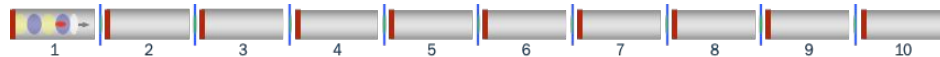
# Science Case: Staging of Laser-Driven Plasma Acceleration

**Goal: deliver & scientifically use** the nation's first exascale systems



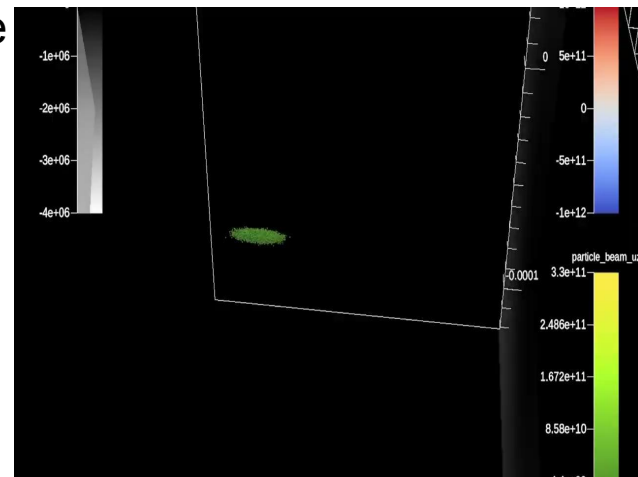
- **ExaFLOP**: a quintillion ( $10^{18}$ ) calculations per second
- ensure ***all*** the necessary pieces are *concurrently* in place

Our DOE science case is in **HEP**, our methods are **ASCR**:  
first 3D simulation of a chain of plasma  
accelerator stages for future colliders



**ECP hardware & software co-design paid off:**

- WarpX is **500x** more performant than the pre-ECP baseline
- We were the **first in ECP** to run at scale on Frontier



**First-of-their-kind platforms:** NERSC (Intel, then Nvidia)→Exascale: OLCF (AMD), ALCF (Intel)

